APLIKASI GRANTT CHART PADA ALGORITMA PENJADUALAN PROSES

(Grantt Chart Application on Scheduling Algorithm Process)

Sri Handayani, April Firman Daru
Program Studi Teknik Informatika Jurusan Teknologi Informasi
Fakultas Teknologi Informasi dan Komunikasi, Universitas Semarang
email : sri@usm.ac.id, firmanusm@gmail.com

Abstract

Operating system is a compulsory subject taken in the Information Technology Student FTIK - USM . In the discussion of the material management process of the operating systems course , there are some scheduling algorithm which processes need to be understood and any existing process scheduling algorithm presented in the form of Grantt Charts . Because of the difficulties experienced by students in understanding the process of scheduling algorithms by an Operating System become researchers inspiration to conduct this study to make an application Grantt Chart.. This study uses a waterfall system development , where the flow will follow the research the phase in waterfall . This application will be built using Visual Basic 6 . The purpose of this research is to produce an application of learning are presented in graphical form to facilitate students in understanding the material scheduling process performed by an operating system so that students can also understand how a computer can do multiple processes simultaneously .

Keywords*: Grantt Chart, Process-scheduling algorithm, Visual Basic 6*

1. PENDAHULUAN

Sistem Operasi merupakan mata kuliah wajib yang diambil mahasiswa Teknik Informatika di FTIK-USM. Mata kuliah ini membahas tentang konsep dasar suatu perangkat lunak yang menjembatani komputer dan pengguna, sehingga memudahkan pengguna untuk mengoperasikannya. Pada materi pembahasan manajemen proses dari mata kuliah Sistem Operasi ini, terdapat beberapa algoritma penjadualan proses yang dapat digunakan. Tiap algoritma penjadualan proses yang dapat digunakan, semua dapat tersaji dalam bentuk grafik *Grantt* (*Grantt Chart*). Dari *Grantt Chart* pula dapat ditentukan waktu tunggu rata-rata tiap proses yang ada dalam prosesor komputer.

Peneliti merasakan pentingnya materi manajemen proses dalam mata kuliah Sistem Operasi sebagai dasar bagi mahasiswa teknik Informatika agar mudah memahami bagaimana sebuah komputer seperti dapat mengerjakan beberapa proses secara bersamaan. Sementara itu dari pengalaman peneliti selama mengampu mata kuliah Sistem Operasi ini, sebagian besar mahasiswa menemui kesulitan menyelesaikan persoalan algoritma penjadualan proses yang diberikan oleh Dosen.

Dari permasalahan tersebut, maka peneliti tertarik untuk membuat suatu aplikasi *Grantt Chart* yang ditujukan untuk membantu mahasiswa agar lebih mudah dalam memahami materi algoritma penjadualan proses tersebut dengan cara membandingkan hasil penggunaan aplikasi *Grantt Chart* dengan hasil perhitungan manual, selain itu juga sebagai sarana peneliti untuk menyajikan materi algoritma penjadualan proses dalam bentuk yang lebih interakif.

2. METODE PENELITIAN

2.1. Jenis Data

Jenis Data yang digunakan oleh peneliti ada dua jenis yaitu :

a. Data Primer, yaitu data-data yang diperoleh peneliti secara langsung, contohnya adalah data hasil nilai mahasiswa selama 2 tahun terakhir dari mata kuliah Sistem Operasi untuk bahasan manajemen proses.
b. Data Sekunder, yaitu data-data yang diperoleh peneliti dari literatur, buku referensi, ataupun dari browsing internet.

2.2. Metode Pengumpulan Data

Metode pengumpulan data yang dilakukan dalam penelitian ini adalah :

a. Observasi: Mengamati kebutuhan *user* (pengguna) yang akan menggunakan aplikasi *Grantt Chart* sebagai alat bantu pembelajaran materi penjadualan proses dalam mata kuliah Sistem Operasi.
b. Studi Pustaka : Mengumpulan literatur pendukung penelitian, baik dari buku referensi ataupun dari *browsing* internet.

2.3. Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan adalah model *waterfall* (Al Bahra Bin Ladjamudin, 2010) yang mempunyai tahapan-tahapan:

1. Analisa Kebutuhan
   Penentuan definisi dari sistem yang diperlukan, penjelasan dan tujuan dari sistem.
2. Perancangan Sistem
   Desain sistem membagi proses dari kebutuhan yang diperlukan ke salah satu perangkat keras atau perangkat lunak. Desain sistem menetapkan arsitektur sistem secara menyeluruh. Desain perangkat lunak melibatkan pengidentifikasian dan pendeskripsian dari sistem beserta relasinya.
3. Implementasi
   Desain dari sistem perangkat lunak diterapkan menjadi suatu aplikasi *Grantt Chart*.
4. Pengujian
   Unit program atau program diintegrasikan dan diuji sebagai satu sistem aplikasi *Grantt Chart* untuk memastikan bahwa seluruh kebutuhan perangkat lunak dan perangkat keras telah terpenuhi setelah pengujian sistem diberikan kepada pengguna dengan menggunakan pengujian perangkat lunak yaitu *Black Box* dan *White Box* sehingga dapat dilakukan perbaikan agar sistem dapat dimanfaatkan secara maksimal sesuai dengan tujuan pembuatan sistem.
5. Penggunaan dan Pemeliharaan
   Instalasi dan pemeliharaan sistem *hardware* dan *software* dilakukan untuk mengembangkan implementasi dari unit sistem. *Software* yang digunakan untuk mengembangkan aplikasi *Grantt Chart* adalah Visual Basic 6.

2.4. Lokasi Penelitian

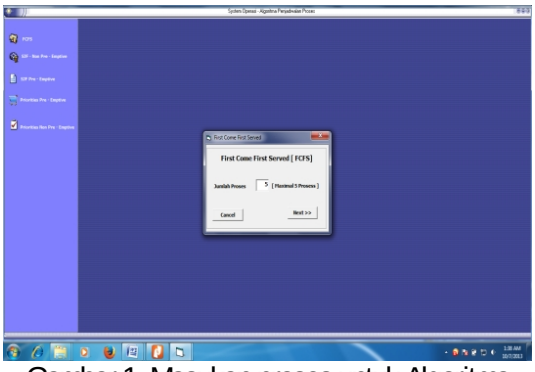
Lokasi penelitian dilakukan di Fakultas Teknologi Informasi dan Komunikasi Universitas Semarang (FTIK-USM) dengan sasaran pengguna hasil penelitian ini adalah mahasiswa Teknik Informatika semester 3 peserta perkuliahan Sistem Operasi.

3. HASIL DAN PEMBAHASAN

Implementasi aplikasi *Grantt Chart* yang telah dibuat adalah:

1. Algoritma FCFS
2. Algoritma SJF Pre-Emptive
3. Algoritma SJF Non Pre-Emptive
4. Algoritma Priorities Pre-Emptive
5. Algoritma Priorities Non Pre-Emptive

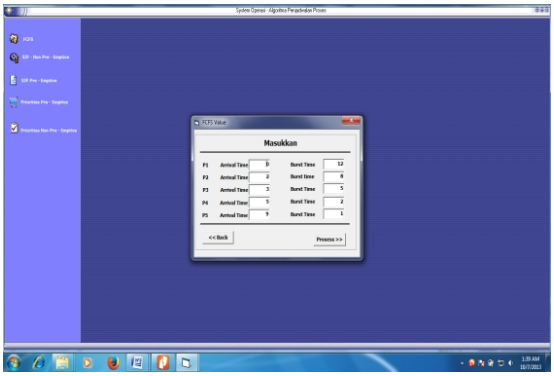
Contoh hasil aplikasi Grantt Chart dari algoritma FCFS



Gambar 1. Masukan proses untuk Algoritma FCFS

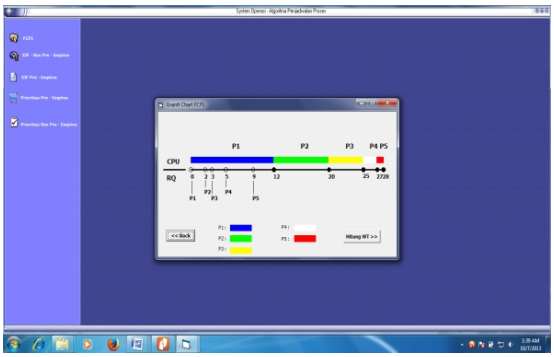
Algoritma FCFS bekerja berdasar proses yang pertama datang akan dilayani terlebih dahulu, tidak peduli burst timenya panjang atau

pendek. Bila sebuah proses sedang dikerjakan , maka akan diselesaikan terlebih dahulu dan baru kemudian proses berikutnya dilayani.Masukan untuk proses dari aplikasi yang dibuat adalah maksimum 5 proses (gambar 1). Pada gambar 2. terlihat masukan yang diberikan user adalah 5. Waktu kedatangan Proses 1=P1 = 0,P2=2,P3=3,P4=5,dan P5=9



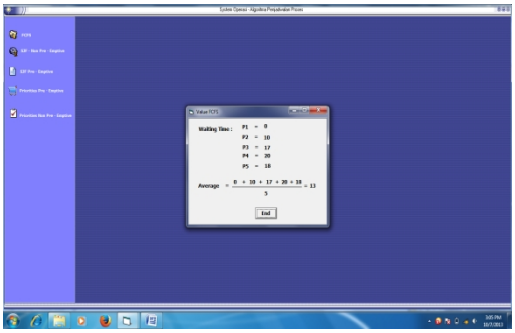
Gambar 2. Pemberian nilai *Arrival Time* dan *BurstTime* untuk 5 inputan proses

Pada gambar 2. *Burst Time* pun diisi oleh *user*. *Burst Time* adalah waktu yang dibutuhkan suatu proses untuk selesai dilayani. Tampak pada gambar 2.Burst Time P1=12,P2=8,P3=5, P4=2, dan P5=1.



Gambar 3. Hasil tampilan *Grantt Chart*

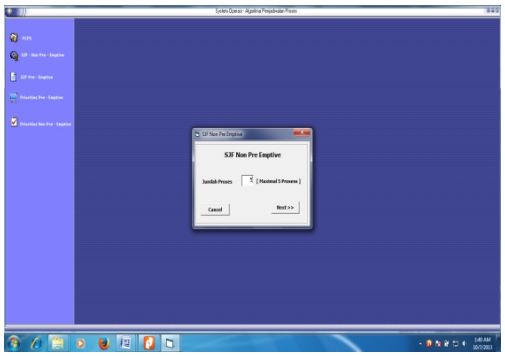
Pada gambar 3. Tampak hasil tampilan *Grantt Chart* untuk 5 masukan menggunakan algoritma FCFS



Gambar 4. Hasil perhitungan waktu tunggu rata-rata tiap proses dengan FCFS

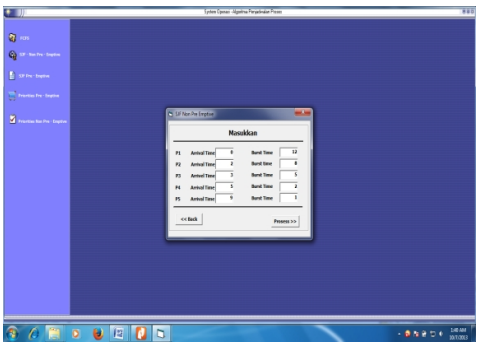
Pada gambar 4, tampak hasil perhitungan waktu tunggu rata-rata proses dengan menggunakan algoritma FCFS. Prinsipnya adalah menghitung selisih waktu suatu proses telah selesai dilayani dengan waktu suatu proses harus berhenti sementara.

Contoh hasil aplikasi Grantt Chart dari algoritma SJF *Non-Pre- Emptive*



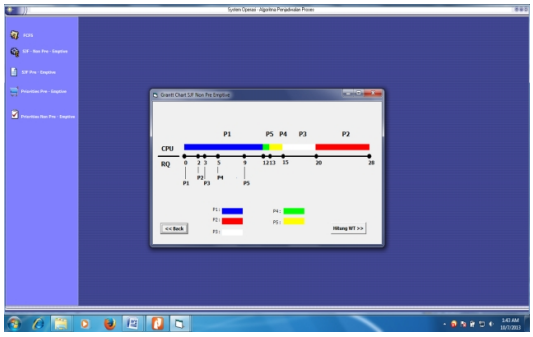
Gambar 5. Masukan proses untuk Algoritma SJF *Non-Pre-Emptive*

Algoritma SJF *Non-Pre-Emptive* mirip dengan FCFS. Hanya saja pada algoritma ini, jika *arrival time* sama, maka proses yang *burst time*-nya lebih pendek akan diproses lebih dulu. Masukan untuk proses dari aplikasi yang dibuat adalah maksimum 5 proses. Pada gambar 6. Waktu kedatangan diisi oleh *user* dengan prinsip, proses 1 memiliki waktu kedatangan (*arrival time*) lebih kecil dari proses ke-2, Proses ke-2 memiliki waktu kedatangan lebih kecil dari proses ke-3, dst. *Burst Time* diisi oleh *user*. *Burst Time* adalah yang dibutuhkan proses untuk selesai dilayani.

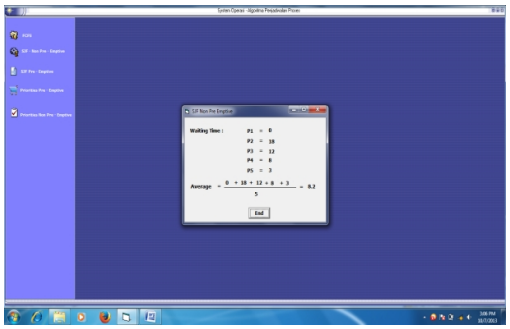


Gambar 6. Pemberian nilai *Arrival Time* dan *Burst* Time untuk 5 inputan proses

Pada gambar 6 Proses 1 datang waktu ke 0, memiliki *burst time* 12. Proses 2 datang waktu ke 2 dengan *burst time* 8, Proses 3 datang waktu ke 3 memiliki *burst time* 5, Proses 4 datang waktu ke 5 memiliki *burst time* 2, dan Proses 5 datang waktu ke 9 *burst time* 1



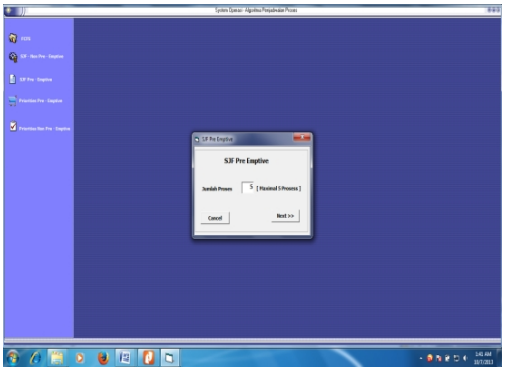
Gambar 7. Hasil tampilan *Grantt Chart* untuk Algoritma SJF *Non Pre-Emptive*



Gambar 8. Hasil perhitungan waktu tunggu rata-rata tiap proses dengan SJF *Non Pre-Emptive*

Pada gambar 8, terlihat perhitungan waktu tunggu rata-rata tiap proses dalam sistem (*execution unit*) prosesor. P1 tidak memiliki waktu tunggu, karena saat P1 datang, P1 langsung dieksekusi (dilayani). P2 memiliki waktu tunggu 18 = 20-2, karena P2 datang waktu ke 2, lalu P2 mulai dieksekusi sampai selesai waktu ke 20. P3 memiliki waktu tunggu 12 = 15-3, karena P3 datang waktu ke 3, lalu P3 mulai dieksekusi sampai selesai waktu ke 15. P4 memiliki waktu tunggu 8 = 13-5,karena P4 datang waktu ke 5, dan P4 mulai dieksekusi sampai selesai waktu ke 13. Dan P5 memiliki waktu tunggu 3 = 12-9 karena P5 datang waktu ke 9 mulai dieksekusi sampai selesai waktu ke 12. Total waktu tunggu rata-rata proses dalam sistem adalah Penjumlahan hasil waktu tunggu tiap proses dibagi jumlah proses yang ada.

Contoh hasil aplikasi Grantt Chart dari algoritma SJF *Pre-Emptive*



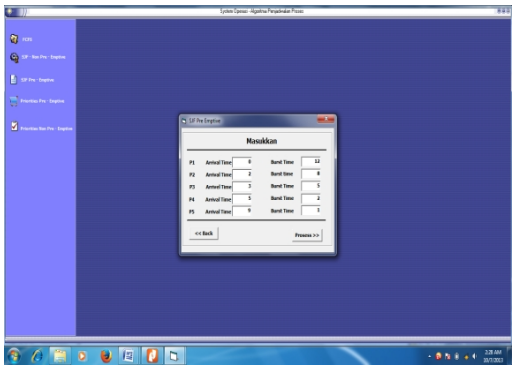
Gambar 9. Masukan untuk algoritma SJF Pre-Emptive

Pada Algoritma ini jika suatu proses sedang dikerjakan kemudian ada proses lain yang datang, di mana *burst time*nya lebih kecil dari sisa waktu proses yang sedang dikerjakan, maka proses lain ini akan dikerjakan terlebih dahulu sementara sisa proses tadi dikembalikan ke *Ready Queue* (RQ).

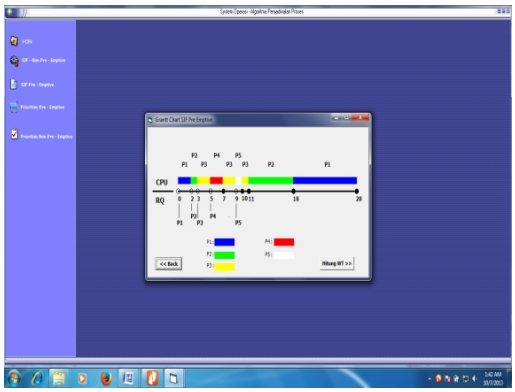
Pada gambar 10. Waktu kedatangan diisi oleh *user* dengan prinsip, proses 1 memiliki waktu kedatangan (*arrival time*) lebih kecil dari proses ke-2, Proses ke-2 memiliki waktu kedatangan dari proses ke-3, dst. *Burst Time* diisi oleh *user*. *Burst Time* adalah yang dibutuhkan proses untuk selesai dilayani.

Pada gambar 10 Proses 1 datang waktu ke 0, memiliki *burst time* 12. Proses 2 datang waktu ke

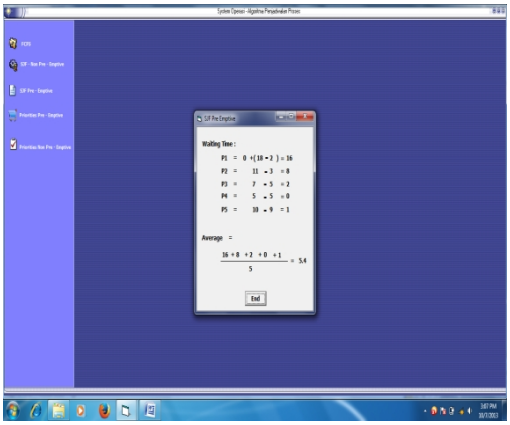
2 dengan *burst time* 8, Proses 3 datang waktu ke 3 memiliki *burst time* 5, Proses 4 datang waktu ke 5 memiliki *burst time* 2, dan Proses 5 datang waktu ke 9 *burst time* 1



Gambar 10. Pemberian nilai *Arrival* Time dan Burst Time



Gambar 11. Hasil Tampilan *Grantt Chart* untuk SJF *Pre-Emptive*



Gambar 12. Hasil perhitungan waktu tunggu rata-rata untuk SJF *Pre-Emptive*

Pada gambar 12, terlihat perhitungan waktu tunggu rata-rata tiap proses dalam sistem (*execution unit*) prosesor. P1 memiliki waktu tunggu 16=0+(18-2), karena P1 datang waktu ke 0, dieksekusi sampai waktu ke 2, pada waktu ke 2 datang pula P2, sehingga di waktu ke 2 , P1 akan dibandingkan dengan P2. Ternyata P2 memiliki *burst time* lebih kecil dari P1, sehingga P2 yang dieksekusi mulai waktu ke 2 (P1 menunggu). P1 mulai dieksekusi lagi waktu ke 18. P2 memiliki waktu tunggu 8 = 11-3, karena P2 datang waktu ke 2, saat itu juga ada P1, karena P2 memiliki *burst time* lebih kecil dari P1, maka P2 dieksekusi, memasuki waktu ke 3, masuk P3, *burst time* P2 dibandingkan dengan *burst time* P3, diperoleh P3 *burst time*nya lebih kecil dari P2, sehingga P3 dieksekusi (P2 menunggu). lalu P2 mulai dieksekusi kembali waktu ke 11.

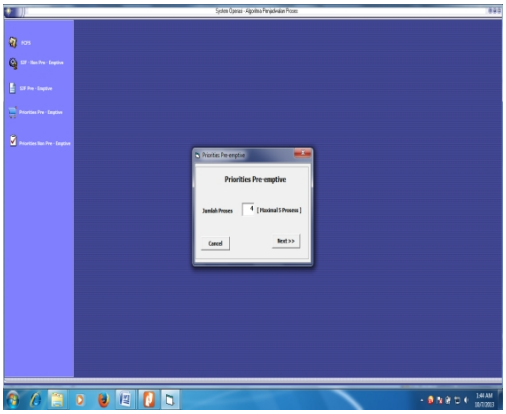
P3 memiliki waktu tunggu 2 =7-5, karena P3 datang waktu ke 3, saat itu juga dalam sistem P1,P2,P3. Dari ke-3 proses yang sudah masuk ternyata P3 memiliki *burst time* paling kecil, karena itu saat P3 datang P3 langsung dieksekusi sampai waktu 5. Saat waktu ke 5, masuklah P4, proses-proses yang sudah masuk telah ada akan dibandingkan *burst time*nya dengan P4. Dari P1,P2,P3,P4, ternyata *burst time* terkecil. Maka P4 akan dieksekusi lebih dulu. (P3 menunggu), P3 kembali dieksekusi waktu ke 7. Karena pada waktu ke 7, P3 adalah proses yang memiliki *burst time* terkecil dibanding P1,P2 (P4 sudah selesai dieksekusi). P4 datang waktu ke 5, dan dieksekusi waktu ke 5. Sehingga P4 tidak memiliki waktu tunggu Dan P5 memiliki waktu tunggu 1= 10-9 karena P5 datang waktu ke 9 mulai dieksekusi waktu ke 10.

Total waktu tunggu rata-rata proses dalam sistem adalah Penjumlahan hasil waktu tunggu tiap proses dibagi jumlah proses yang ada.

Contoh hasil aplikasi Grantt Chart dari algoritma *Priorities Pre- Emptive*

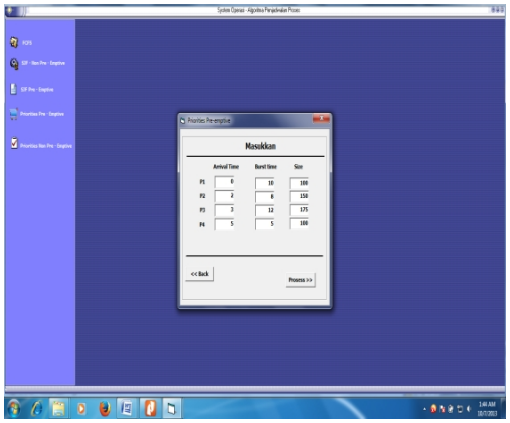
Dengan algoritma *priority schedulling*, CPU akan mengerjakan prioritas yang lebih tinggi terlebih dahulu, dan jika beberapa proses memiliki prioritas yang sama, maka akan

digunakan FCFS. Prioritas dari proses terlihat dari *size* proses masing-masing



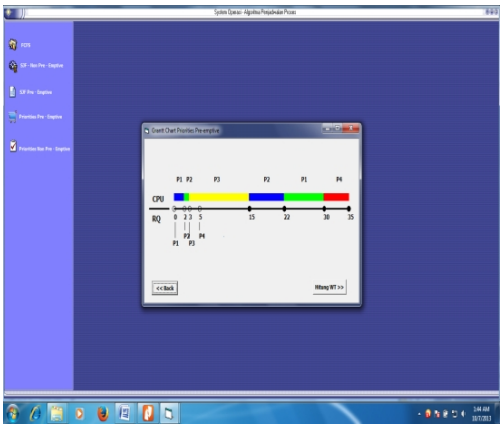
Gambar 13. Masukan 4 proses pada algoritma *Priorities Pre-Emptive*

Pada gambar 14. Waktu kedatangan diisi oleh *user* dengan prinsip,proses 1 memiliki waktu kedatangan (*arrival time*) lebih kecil dari proses ke-2, Proses ke-2 memiliki waktu kedatangan dari proses ke-3, dst. *Burst Time* dan *Size* diisi oleh *user*. *Burst Time* adalah yang dibutuhkan proses untuk selesai dilayani.Sementara *Size* menunjukkan prioritas yang melekat dari setiap proses.

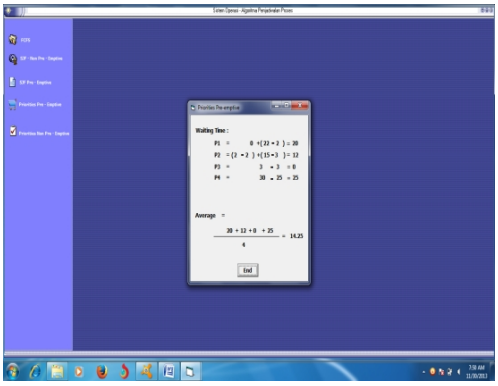


Gambar 14. Pemberian nilai *Burst time, Arrival Time*, dan *Size*

Pada gambar 14 Proses 1 datang waktu ke 0, memiliki *burst time* 10, *size* 100. Proses 2 datang waktu ke 2 dengan *burst time* 8, *size* 150 Proses 3 datang waktu ke 3 memiliki *burst time* 12, *size* 175 Proses 4 datang waktu ke 5 memiliki *burst time* 5, dan *size* 100



Gambar 15. Hasil Tampilan *Grantt Chart* untuk algoritma *Priorities Pre- Emptive*



Gambar 16. Perhitungan waktu tunggu rata-ratanya.

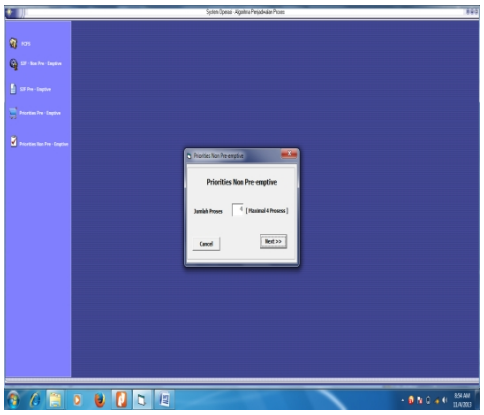
P1 memiliki waktu tunggu 20 = 0 + (22-2) karena datang waktu ke 0 kemudian dieksekusi,sampai waktu ke 2, karena pada waktu ke 2 datang P2 yang memiliki *size* lebih besar dari P1, P2 dieksekusi waktu ke 2,P1 menunggu, dan kembali dieksekusi waktu ke 22.

P2 datang waktu ke 2, pada saat terdapat P1 yang sedang berjalan. Karena P2 memiliki *size* lebih besar dari P1, maka P2, akan dieksekusi dulu.P1 menunggu. P2 berjalan sampai waktu ke 3, pada saat itu pula masuk P3 yang *size*nya lebih besar dari P2, maka P2 berhenti dulu karena P3 yang akan dieksekusi. P2 akan kembali dilayani waktu ke 15 sampai selesai. P3 datang waktu ke 3 dan dilayani, namun pada waktu ke 5 datang P4,prioritas P4 lebih kecil dari P3 sehingga P3 tetap dilayani sampai selesai,sehingga P3 tidak memiliki waktu tunggu.Selanjutnya setelah P3 selesai, seluruh

proses telah masuk, proses yang akan dilayani selanjutnya adalah proses yang memiliki size terbesar setelah P3. Dari P1,P2,P4 yang belum selesai dilayani, P2 memiliki *size* (prioritas) terbesar, karena itu P2 akan dilanjutkan dilayani sampai selesai di waktu ke 22.Setelah waktu ke 22, P1 dan P4 memiliki *size* yang sama, namun karena P1 yang lebih dulu datang dari P4, maka prinsip FCFS yang akan dikerjakan sehingga P1 yang akan dilayani sampai selesai, dan terakhir P4 yang akan dilayani sampai selesai.

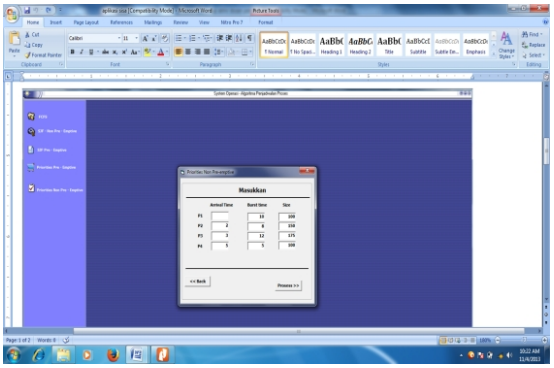
Total waktu tunggu rata-rata proses dalam sistem adalah Penjumlahan hasil waktu tunggu tiap proses dibagi jumlah proses yang ada.

Contoh hasil aplikasi Grantt Chart dengan algoritma *Priorities Non Pre-Emptive:*



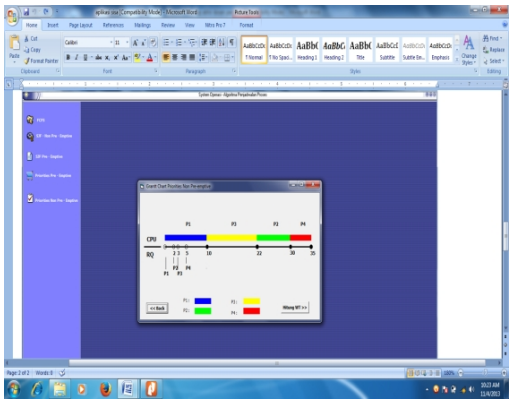
Gambar 17. Masukan 5 proses untuk algoritma *Priorities Non Pre-Emptive*

Dengan algoritma *Priorities Non Pre-Emptive*, CPU akan menyelesaikan proses yang pertama kali dating lalu mengerjakan prioritas yang lebih tinggi terlebih dahulu, dan jika beberapa proses memiliki prioritas yang sama, maka akan digunakan FCFS. Prioritas dari proses terlihat dari *size* proses masing-masing.

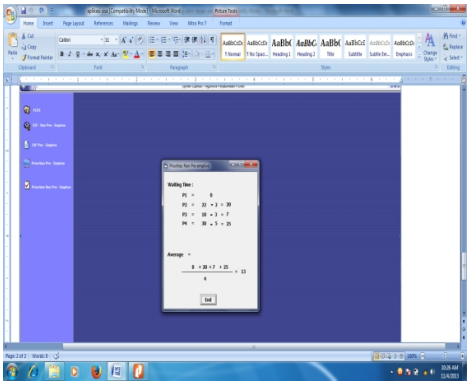


Gambar 18. Masukan untuk 5 proses dalam algoritma *Priorites Non Pre-Emptive*

Pada gambar 18 Proses 1 datang waktu ke 0, memiliki *burst time* 10, *size* 100. Proses 2 datang waktu ke 2 dengan *burst time* 8, *size* 150 Proses 3 datang waktu ke 3 memiliki *burst time* 12, *size* 175 Proses 4 datang waktu ke 5 memiliki *burst time* 5, dan *size* 100



Gambar 19. Hasil tampilan *Grantt Chart* dengan algoritma *Priorities Non Pre-Emptive*



Gambar 20. Perhitungan waktu tunggu rata-rata tiap proses dalam sistem

Pada Gambar 20, terlihat P1 datang lebih dulu langsung dilayani sampai selesai sesuai dengan *Burst time* 10, sehingga jatah waktu proses yang diberikan sistem pada P1 adalah 10. Di waktu ke 10, seluruh proses (P2,P3,P4) sudah masuk. Setelah P1 selesai selanjutnya yang berjalan adalah proses yang memilik *size* terbesar setelah P1 yaitu P3 yang memiliki *size* 175. P3 dilayani sampai selesai mulai waktu ke 10 sampai waktu ke 22 = 10 + 12 . 12 adalah *burst time* atau jatah waktu proses P3, sehingga P3 memiliki waktu tunggu 10-3=7 (3 adalah waktu kedatangan P3). Selanjutnya di waktu ke 22 yang akan dilayani adalah P2 karena P2 memiliki *size* terbesar setelah P3 yaitu 150. P2 dilayani sampai selesai waktu ke 30= 22+8), 8 adalah jatah waktu proses/*burst time* P2. P2 memiliki waktu tunggu 22-2= 20, yaitu P2 dilayani sampai selesai dari waktu ke 22 dan di waktu ke 2 P2 datang langsung berhenti. Akhirnya tinggal P4 yang belum dilayani, mulai waktu ke 30, P4 dilayani sampai selesai di waktu ke 35= 30+5, P4 memiliki waktu tunggu 30-5=25. (30 adalah waktu P4 dilayani sampai selesai, dan 5 adalah waktu kedatangan P4, P4 datang langsung berhenti). Total waktu tunggu rata-rata proses dalam sistem adalah Penjumlahan hasil waktu tunggu tiap proses dibagi jumlah proses yang ada.

4. KESIMPULAN DAN SARAN

4.1. Kesimpulan

1. *Grantt Chart* yang dihasilkan di aplikasi ini, penggambaran grafik menggunakan warna yang berbeda untuk mewakili proses yang berbeda.
2. Tampilan *chart* berwarna dalam aplikasi ini diharapkan membantu *user* untuk memahami materi strategi penjadualan proses.
3. Maksimum dari inputan untuk aplikasi ini adalah 5 proses, sementara untuk Algoritma *Priorities* peneliti inputan yang dapat diimplementasikan peneliti maksimum 4 inputan.
4. Grafik yang dibuat hanya untuk 5 algoritma yaitu : FCFS,SJF non Pre-Emptive, SJF Pre - Emptive, Priorities Pre - Emptive, dan Priorities non Pre-Emptive.
5. Aplikasi ini tidak menggunakan *database*, sehingga tiap inputan, proses perhitungan, dan outputan grafik yang dihasilkan tidak tersimpan di sistem.
6. Aplikasi bantu ajar ini tidak menggunakan *database*, dan karena aplikasi ini menampilkan gambar grafis, maka untuk aplikasi *Grantt Chart* ini termasuk aplikasi yang membutuhkan memori, dengan kapasitas sebesar 5,53 Mb
7. Aplikasi bantu ajar ini masih bersifat statis, dimana input yang diberikan baru dapat ditentukan peneliti.

4.2. Saran

1. Aplikasi ini akan diuji cobakan ke mahasiswa, untuk mengetahui perbedaan pemahaman materi Algoritma Penjadualan Proses dengan menggunakan alat bantu aplikasi Grantt Chart dengan pemahaman materi Algoritma Penjadualan Proses tanpa alat bantu ajar.
2. Untuk menghitung waktu tunggu rata-rata proses dalam sistem, peneliti masih perlu memikirkan bagaimana cara menyajikannya di aplikasi, agar user lebih memahami perhitungan yang ada di setiap algoritma.
3. Perlu penelitian lebih lanjut lagi agar aplikasi Grantt Chart ini memiliki kapasitas memori yang lebih kecil dari 5,53 Mb.
4. Perlu penelitian lebih lanjut lagi agar aplikasi Grantt Chart ini menjadi aplikasi dinamis yang inputnya bisa diubah sesuai keinginan user.

DAFTAR PUSTAKA

1. Abas Ali Pangeran,2008, Sistem Operasi, Andi Offset
2. Bambang Hariyanto,2009,Sistem Operasi, Informatika
3. Kusnadi,Kusworo Anindito,Y.Sigit Purnomo,WP,2009,Sistem Operasi,Andi Publisher
4. Al Bahra Bin Ladjamudin, 2010, Rekayasa Perangkat Lunak, Graha Ilmu
5. Sabar & Yuswanto 2007, Panduan Lengkap Pemrograman Visual Basic 6